



Universidad  
Politécnica  
de Madrid

# ETSI SISTEMAS INFORMÁTICOS

**Proyecto de Fin de Grado**

**Desarrollo de un sistema de información de servicios de líneas  
de autobuses**

**Grado en Ingeniería del Software**

**Carlos Aldeguez Silvero**

**Miguel Ángel Díaz Martínez**

**Curso 2016 - 2017**



## Resumen

Con el auge de las tecnologías móviles en los últimos años han aflorado diversas aplicaciones, tanto móviles como web, en las que el usuario puede consultar los horarios del transporte público, sea autobús, tren o metro. No obstante, los usuarios de las líneas de autobús L411 (Perales del Río - Legazpi) y L4 (Perales del Río - Getafe) no disponen de ningún servicio donde poder consultar la información de sus autobuses.

Con este Proyecto de Fin de Grado se busca paliar esta carencia desarrollando una aplicación móvil para dispositivos Android que permita la visualización tanto de los horarios como de las paradas de las líneas de autobús L411 y L4.

Además se diseñará e implementará una base de datos en un servidor que servirá una API consumible desde cualquier dispositivo capaz de realizar peticiones HTTP para aumentar la escalabilidad y facilitar la actualización de las aplicaciones de los usuarios.



## **Abstract**

With the rise of mobile technologies in recent years there have emerged diverse applications, both mobile and web, in which the user can consult the schedules of public transport, whether bus, train or subway. However, users of the bus lines L411 (Perales del Rio - Legazpi) and L4 (Perales del Rio - Getafe) have no service where they can consult information related to their buses.

This End-of-Grade Project seeks to alleviate this lack by developing a mobile application for Android devices that allows the visualization of both the schedules and stops of bus lines L411 and L4.

In addition, a database will be designed and implemented on a server that will serve a consumable API from any device capable of performing HTTP requests to increase scalability and facilitate the updating of users' applications.



## **Agradecimientos**

A mis padres y hermana, por apoyarme incondicionalmente siempre que han podido, sin ellos no podría haber llegado hasta aquí.

A Miriam, por haberme acompañado durante estos cinco años y haberme ofrecido su ayuda.

A Andrei, por haber estado siempre a mi lado, en lo bueno y en lo malo.

Y principalmente, a todas aquellas personas que en algún momento dudaron que pudiera llegar hasta aquí, sin ellos ésto no sería tan satisfactorio.





## Tabla de contenido

Resumen.....	3
Abstract .....	5
Introducción .....	12
Motivación .....	12
Objetivos .....	13
Estado del arte .....	15
Tecnologías empleadas .....	17
Android Studio.....	17
Java .....	17
XML .....	18
JSON .....	18
PHP .....	19
API Google Maps .....	19
SQLite .....	20
Hostinger.....	20
phpMyAdmin .....	20
Git .....	21
SourceTree .....	21
Postman.....	21
REST .....	22
Metodología de trabajo .....	23
Metodología Agile .....	23
Kanban.....	24
Diagramas .....	28
Casos de uso .....	28
Modelo Entidad Relación.....	30
Clases .....	31
Despliegue .....	33
Especificación de requisitos.....	34
Implementación de la metodología.....	36
Sprints .....	39
Sprint 1.....	39
Sprint 2.....	44
Sprint 3.....	46

Sprint 4.....	47
Metodología de pruebas.....	49
Manual de usuario.....	50
Conclusiones y líneas de continuación.....	55
Conclusiones.....	55
Líneas de continuación .....	56
Bibliografía.....	57



## Introducción

### Motivación

Hoy en día es casi imposible entender la vida moderna sin nuestros teléfonos móviles, esos dispositivos que se han convertido en nuestro fiel acompañante, permitiéndonos estar comunicados, informados e incluso controlados.

Debido a este aumento del uso de los mismos han surgido múltiples aplicaciones que buscan aportar cierta información al usuario para lograr facilitarle la vida y, éste proyecto, es una de ellas.

En este proyecto se busca realizar una aplicación que permita al usuario consultar, de una forma clara y sencilla, los horarios y paradas de dos líneas de autobús que, hasta ahora, no podían consultar en ninguna de las demás aplicaciones de esta índole ya existentes.

## Objetivos

El objetivo principal es el desarrollo de un sistema de información que muestre los datos (horarios y paradas) de las líneas L411 (Perales del Rio - Legazpi) and L4 (Perales del Rio - Getafe).

Para ello desarrollaremos dos partes claramente diferenciadas que compondrán el total de nuestra aplicación:

1. Se tendrá, por un lado, una base de datos alojada en un servidor con la que nos comunicaremos a través de una API, logrando así que nuestro sistema goce de una mayor escalabilidad y facilitando la actualización de la base de datos de las aplicaciones móviles.
2. Por otra parte, tendremos una aplicación móvil para el sistema operativo Android con la que consumiremos la API anteriormente mencionada, almacenaremos los datos en una base de datos propia y los mostraremos en pantalla. Al almacenarlos en la propia aplicación, logramos que el usuario pueda acceder a ellos sin necesitar conexión a internet, lo que aportará una mayor usabilidad al sistema.

Con todo esto, tendremos una aplicación Android que cumplirá con los objetivos establecidos pudiendo, además, realizar una aplicación con la misma funcionalidad para otros sistemas operativos móviles como pueden ser iOS o Windows Phone sin necesidad de volver a desarrollar la parte del servidor ya que podremos acceder a ésta utilizando la API ya realizada.



## Estado del arte

A la hora de desarrollar una aplicación para estos smartphones podemos optar por hacerla nativa, híbrida o web.

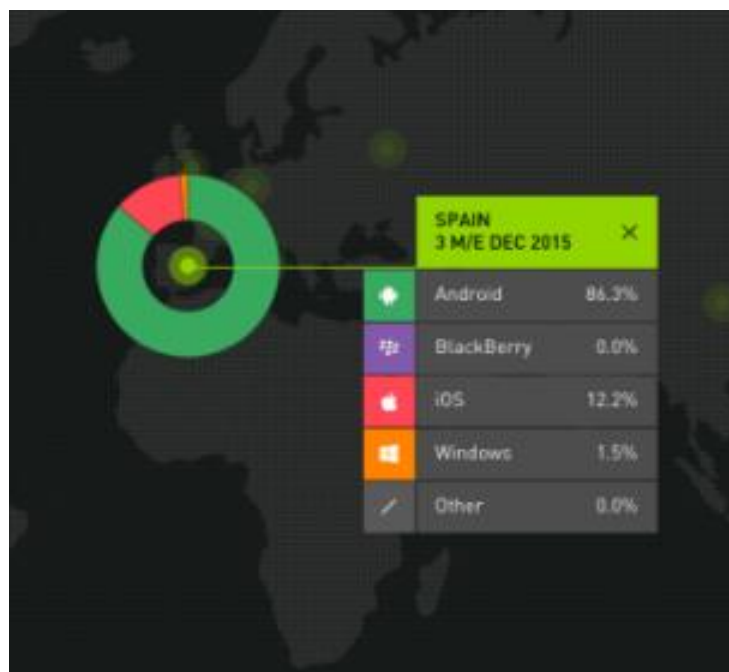
Las aplicaciones nativas son desarrolladas utilizando el lenguaje de programación nativo del dispositivo (Swift en iOS, Java en Android y .Net en Windows Phone). Éste tipo de aplicaciones tiene la ventaja de permitirnos acceder a todas las características del dispositivo y lograr un mayor rendimiento.

Las aplicaciones híbridas son desarrolladas con HTML5, CSS y JavaScript y desplegadas dentro de un contenedor nativo como Phonegap o Cordova que nos brinda acceso a las características del dispositivo de forma independiente al sistema operativo. Estas aplicaciones tienen como principal ventaja su portabilidad.

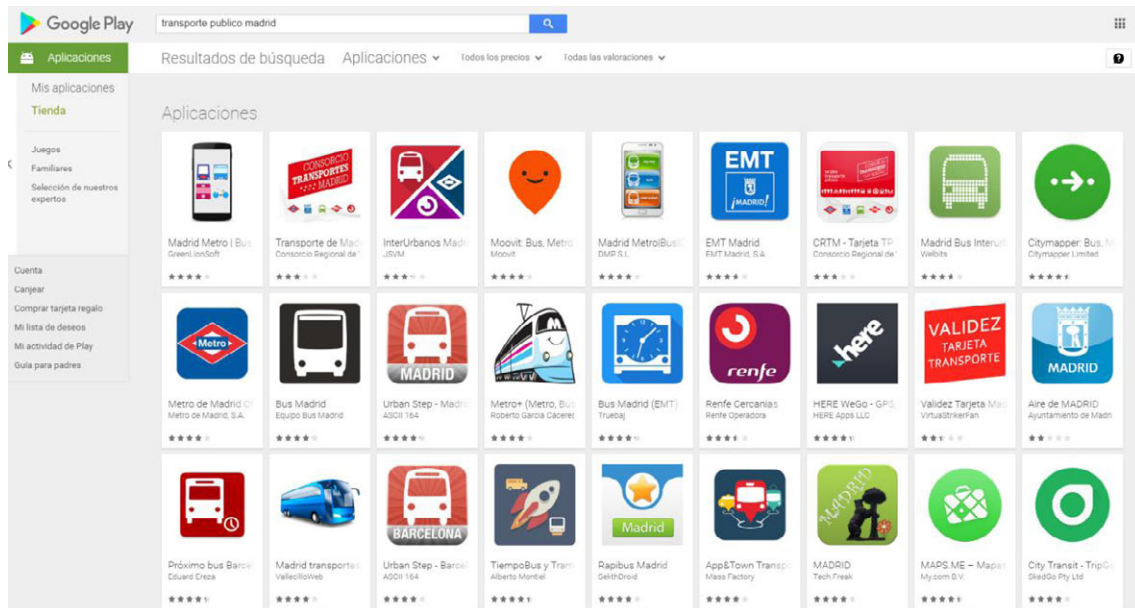
Las aplicaciones web son desarrolladas en HTML5, CSS3 y JQuery y deben ser optimizadas para lograr una correcta visualización desde el Smartphone. Para acceder a ellas debemos hacerlo a través de un navegador, lo que nos obliga a utilizar una conexión a internet, y no tendremos acceso a varias características del dispositivo.

Para el desarrollo de este proyecto se decidió realizar una aplicación nativa por su mayor facilidad de desarrollo, rendimiento y su uso sin conexión a internet.

Una vez tomada esta decisión teníamos tres opciones principales: iOS, Android o Windows Phone. Entre estos tres, decidimos elegir Android por el número de usuarios que lo utilizan en nuestro país, lo cual nos permitirá llegar a un mayor número de usuarios.



Al comprobar qué aplicaciones disponibles en la Google Play Store ofrecían ya este servicio, nos encontramos un gran número de resultados.



Todas ellas tienen un gran número de descargas siendo la mayor Moovit, con más de 450 mil descargas.



No obstante, tras haber probado un gran número de las mismas, observamos que ninguna ofrece información sobre las dos líneas de autobuses que tratamos en este proyecto, lo que nos hace ofrecer un servicio novedoso al usuario de estos medios de transporte.



## Tecnologías empleadas

### Android Studio

Android Studio es el IDE oficial para desarrollo Android, basado en IntelliJ IDEA de JetBrains, está disponible de manera gratuita. Está diseñado para permitir acelerar el desarrollo de las aplicaciones y ayudar a construir aplicaciones de calidad para todos los dispositivos. Características:

- Soporte para el desarrollo basado en Gradle.
- Renderización en tiempo real.
- Sistema de refactorización rápido y sencillo.
- Herramientas de análisis de código que detectan posibles problemas en la aplicación. (Lint)
- Plantillas prediseñadas que facilitan la creación de las aplicaciones.

### Java

James Gosling, Mike Sheridan y Patrick Naughton iniciaron el proyecto del lenguaje Java en 1991. Sun Microsystems lanzó la primera implementación pública de Java en 1995.

Java es un lenguaje de programación basado en clases, orientado a objetos y concurrente, y está específicamente diseñado para tener las menores dependencias posibles. Su intención es permitir a los desarrolladores de aplicaciones “escribir una vez, ejecutar en cualquier lugar”, lo que significa que el código compilado Java puede ser ejecutado en todas las plataformas que soporten Java sin necesidad de volver a compilarlo.

Las aplicaciones de Java son generalmente compiladas a bytecode que puede ser ejecutado en cualquier máquina virtual de Java (JVM) sin importar la arquitectura del ordenador.

En Android los encargados de ejecutar los programas creados en Java son las máquinas virtuales Dalvik y ART.

## XML

XML, que proviene de las siglas en inglés *eXtensible Markup Language*, es un lenguaje de marcas que define una serie de reglas para codificar documentos en un formato que sea legible tanto para las máquinas como para los humanos.

Los objetivos de XML hacen énfasis en la simplicidad, generalidad y usabilidad a través de internet. Aunque el diseño de XML se centra en documentos, es ampliamente usado para la representación de estructuras de datos usadas en web services.

En Android son utilizados para el diseño de la interfaz gráfica, además de para otros casos, como pueda ser la descripción de estilos o variables constantes que son utilizadas en todo el proyecto, como textos, colores o fuentes de texto.

## JSON

JSON, siglas de *JavaScript Object Anotation* es un formato de texto ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript aunque hoy, debido a su amplia adopción como alternativa a XML, se considera un formato de lenguaje independiente.

JSON se caracteriza por reducir el tamaño de los archivos y el volumen de datos que es necesario transmitir frente a otros estándares como XML. Por ello JSON fue adquiriendo popularidad hasta convertirse en un estándar. Esto no significa que XML haya dejado de utilizarse. En la actualidad se utiliza tanto XML como JSON para el intercambio de datos. Utilizar uno u otro depende de las circunstancias y de las preferencias que en cada momento se determinen.

JSON permite la utilización de distintos tipos de datos tales como números, cadenas, booleanos, null, vectores y objetos.

## PHP

PHP, que proviene de *PHP Pre Hypertext –processor*, es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página web resultante.

Características:

- Orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos.
- Tiene capacidad de conexión con la mayoría de motores de bases de datos usados actualmente.
- Sus variables no requieren definición de tipo aunque se pueden evaluar por el tipo que manejan en tiempo de ejecución.

## API Google Maps

Google nos ofrece diversas APIs para utilizar Google Maps en nuestra aplicación, dependiendo de si es Android, iOS o Web estando las primeras disponibles a través de los servicios de Google Play, de forma que nuestra aplicación podrá conocer la ubicación del usuario, incluir mapas, encontrar sitios cercanos y muchas más funciones de este servicio de Google.

## SQLite

SQLite es un sistema gestor de bases de datos relacionales. SQLite se enlaza con el programa pasando a formar parte de este, no es un proceso independiente con el que el programa se comunica.

El programa que hace uso de SQLite utiliza su funcionalidad por medio de subrutinas y funciones permitiendo así reducir la latencia en el acceso a la base de datos ya que las llamadas que se realizan son más eficientes.

Android incorpora de serie todas las herramientas necesarias para la creación y gestión de bases de datos con SQLite

## Hostinger

Hostinger es un proveedor de servicios de hosting que además ofrece otros servicios como constructor de sitios web, compra de dominions, Cloud VPS y un largo etcétera con una variedad de tarifas basadas en tus necesidades.

Los servicios que utilizaremos son el hosting web para obtener la base de datos SQL que ofrece gratuitamente, la compra de un dominio y el administrador de archivos, donde subiremos la API para acceder a la base de datos. Además, tienen integrado phpMyAdmin , lo que nos simplificará la creación, alimentación y mantenimiento de nuestra base de datos.

## phpMyAdmin

phpMyAdmin es una herramienta escrita en PHP para manejar la administración de MySQL a través de páginas web. Actualmente puede crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios y exportar datos en varios formatos.

## Git

Git es un sistema de control de versiones que es usado en su mayor parte para el desarrollo de software. Es un sistema de control distribuido de versiones, que hace énfasis en la velocidad e integridad de los datos y en el apoyo para la distribución de flujos de trabajo no lineales.

### Características

- Apoyo al desarrollo no lineal.
- Desarrollo distribuido.
- Compatibilidad con sistemas y protocolos ya existentes.
- Manejo eficiente de proyectos de gran tamaño.
- Recolector de basura integrado.

## SourceTree

SourceTree es una aplicación de escritorio que permite gestionar los aspectos del repositorio de los proyectos de software, tales como usuarios, ramas, permisos, ver y hacer commits, el código, mostrar estadísticas y muchas características más.

## Postman

Postman es un cliente Rest que permite de manera rápida y fácil realizar peticiones a web services o API, y obtener la respuesta en diferentes formatos. Se usa normalmente a la hora de desarrollar para agilizar el proceso de llamadas y respuestas a los web services.

### Características:

- Histórico de las peticiones realizadas.
- Permite crear peticiones rápidamente.
- Funciones de ayuda para las autenticaciones.
- Personalización con scripts.
- Framework para hacer tests.

## REST

REST o representational state transfer es un tipo de arquitectura que se basa en componentes, conectores y datos en un sistema distribuido. REST es la arquitectura que da soporte a la web actualmente. Se diseñan servicios web que se centran en los recursos del sistema, incluyendo el tipo de acceso al estado de los recursos y como son transferidos por HTTP.

El término REST fue acuñado en 2000 y se basa en cuatro principios básicos. Es capaz de transferir XML, JSON o ambos, expone las URIs con formato de directorios, no mantiene el estado y utiliza métodos HTTP de manera explícita.

En referencia a los métodos de HTTP, asocio los métodos de HTTP con las operaciones clásicas de alta, baja, modificación y consulta. En referencia al no mantenimiento del estado, quiere decir que en cada petición se provee toda la información requerida para comprender la misma, ni el cliente ni el servidor necesitan recordar estados entre comunicaciones anteriores.

Con esto se consigue delegar la responsabilidad del mantenimiento del estado al cliente, el servidor solo será responsable de generar las respuestas y proporcionar una interfaz adecuada para que el cliente pueda mantener el estado de la aplicación de manera adecuada.

Otra característica interesante es que el número de solicitudes no afecta al funcionamiento del servicio REST, puesto que todas las peticiones son independientes.

Gracias a este tipo de arquitectura, el sistema en cierto modo está preparado de antemano para soportar múltiples usuarios y dispositivos de diversas plataformas, siempre y cuando haya un cliente adecuado que sea capaz de aprovecharse de la arquitectura REST.

## Metodología de trabajo

### Metodología Agile

El desarrollo ágil se basa en un desarrollo iterativo e incremental en el cual los requisitos evolucionan con el tiempo según las distintas necesidades del proyecto. Aunque no todas las metodologías ágiles siguen el mismo patrón, todas cuentan con unos ciclos definidos que hay que completar antes de pasar a la siguiente iteración. Estos ciclos tendrán una duración predefinida, que será pactada con el cliente.

En una primera etapa se definirán los requisitos del proyecto, se priorizarán, y se creará un calendario para las iteraciones. Esta planificación no tiene que ser muy estricta ya que uno de los puntos fuertes de este tipo de metodologías es la habilidad de responder rápidamente a los cambios que puedan surgir a lo largo del proyecto.

En la segunda etapa nos encargaremos de la especificación de los requisitos por parte del cliente. Estos requisitos pueden cambiar según avance el proyecto, cambios que deberán ser bienvenidos.

Una vez hemos definidos los requisitos deberemos realizar la implementación de esos requisitos en el proyecto, manteniendo reuniones entre los distintos equipos de desarrollo para poder conocer el estado del proyecto.

En la última fase se deberá testear el software desarrollado para comprobar que todo funcione correctamente.

Con esto se consigue aumentar la eficiencia de los equipos, ya que se tiene más en cuenta la parte del desarrollo del proyecto que la generación de una documentación exhaustiva, y por lo tanto se podrá reaccionar mejor ante los cambios de requisitos que vaya presentando el proyecto mientras que en una metodología tradicional los requisitos estarían definidos desde un primer momento y para cambiarlos habría que volver a realizar toda la documentación desde el principio.

## Kanban

Kanban significa “papel” o “cartel” en japonés, los cuales son unos elementos imprescindibles en esta metodología. Esta metodología surgió en Toyota con el fin de poder dividir el proceso de fabricación de sus coches en fases las cuales debían estar completadas antes de pasar a la siguiente, para así poder conseguir un producto de mayor calidad.

Más tarde, David J. Anderson adaptó este método al desarrollo de software, ya que al igual que en la fabricación de automóviles, éste también cuenta con fases claramente diferenciadas, equipos de trabajo y el requisito imprescindible de que todo funcione correctamente antes de pasar a la siguiente fase.

Los objetivos de este método son dos: crear un producto de mayor calidad y evitar los cuellos de botella que surgen en los proyectos cuando se le da más importancia a la rapidez que a la calidad del producto. Para esto, existen cuatro principios básicos en Kanban:

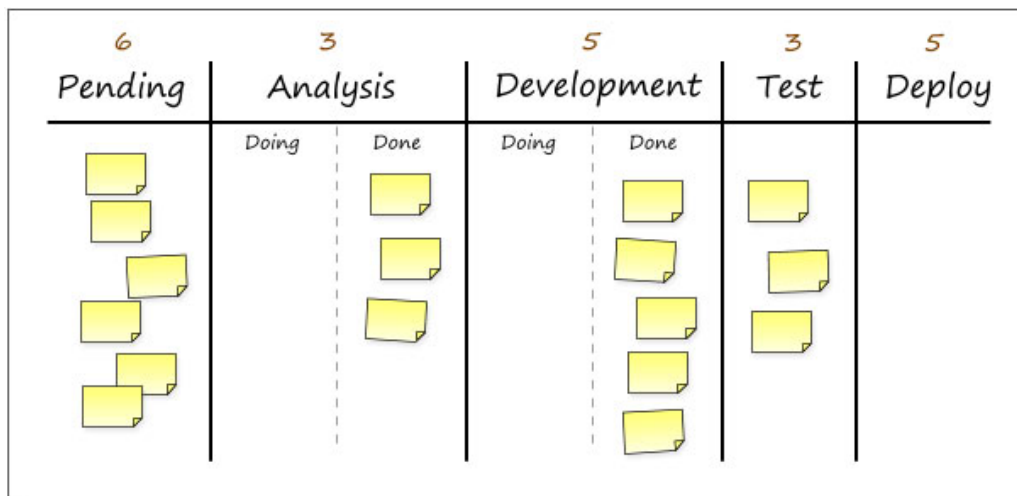
- Empieza con lo que tienes ahora
- Acepta el cambio
- Respeta el proceso en curso, los roles y las responsabilidades de cada rol
- Liderazgo en todos los niveles

A parte de estas reglas, cualquier sistema que aplique correctamente este método deberá contar con estos elementos:

- Visualización del flujo de trabajo: para poder conocer las fases del proyecto o los equipos que trabajan en cada una, suele utilizarse un panel con tarjetas que definan cada una de las tareas, dividiéndolas en columnas que representan las distintas fases del proyecto.
- Limitación del trabajo en curso: con el fin de no dejar tareas sin acabar, se deberá acabar cualquier tarea empezada antes de empezar las demás.



- Gestión del flujo: controlar el flujo de trabajo correctamente para poder detectar cualquier problema en éste.
- Establecimiento claro de las reglas del proceso: para poder aplicar bien Kanban, todos deberán entender su funcionamiento y reglas, por lo tanto que todos sepan cuál es su trabajo y cómo hacerlo es fundamental para el proyecto.
- Mejora en equipo: cualquier cambio deberá acordarse en equipo, para que no existan problemas al implementarlo junto al resto del proyecto.



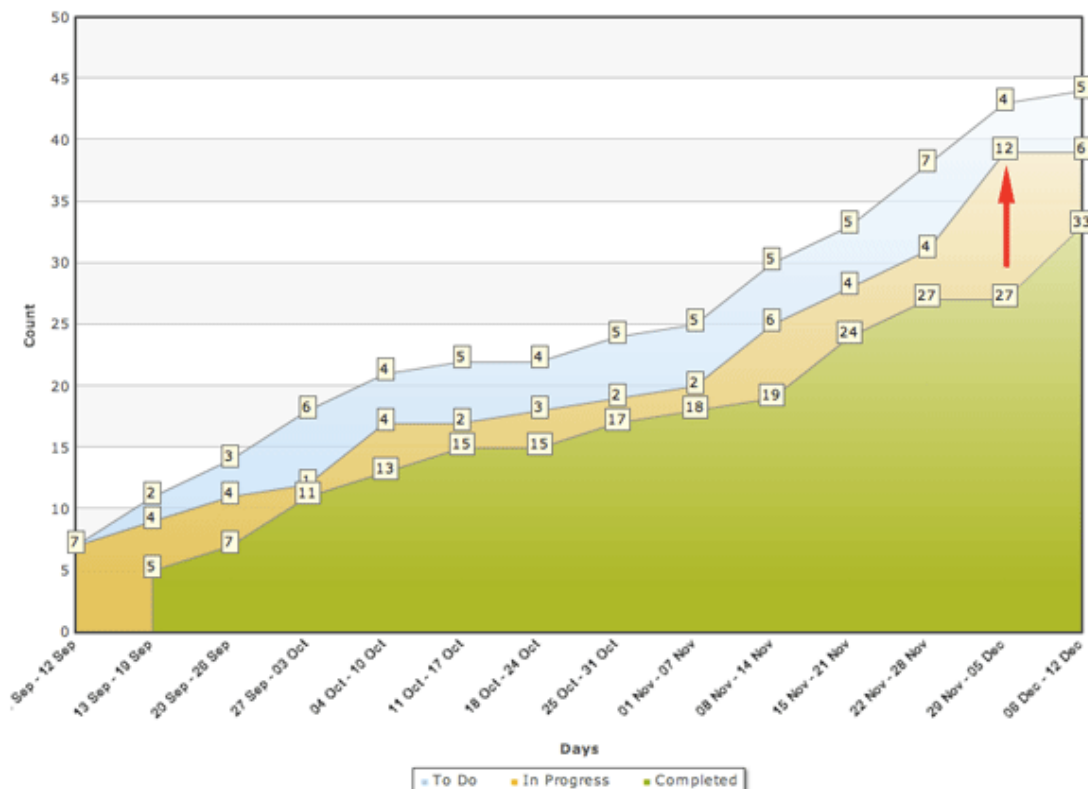
En la figura de arriba podemos ver un flujo de trabajo con Kanban. Las columnas existentes (pendientes, análisis, desarrollo, test y despliegue) serán las distintas fases del proyecto, y los post it serán las distintas tareas que se necesitan completar.

Estas tareas deben pasar de una fase a otra cuando están completamente acabadas. Kanban tiene una gran diferencia con respecto a otras metodologías que utilizan tableros para repartir la carga de trabajo.

Kanban define un número máximo de tareas en cada fase del proyecto, por lo tanto, el número de tarjetas en desarrollo de cada fase del tablero no deberá ser mayor que el número máximo de tareas definido.

Con esta mejora, conseguiremos que los equipos no estén sobrecargados y puedan trabajar en sus tareas correctamente para que el producto sea desarrollado satisfactoriamente y no con rapidez para aligerar la carga de trabajo de los equipos.

Si alguna tarea ya está acabada y necesita pasar a la siguiente fase habrá que tener el número máximo de tareas en cuenta, y si está completo será recomendable que los demás equipos ayuden a rebajar la carga de trabajo en esa fase para poder continuar con normalidad el proyecto.



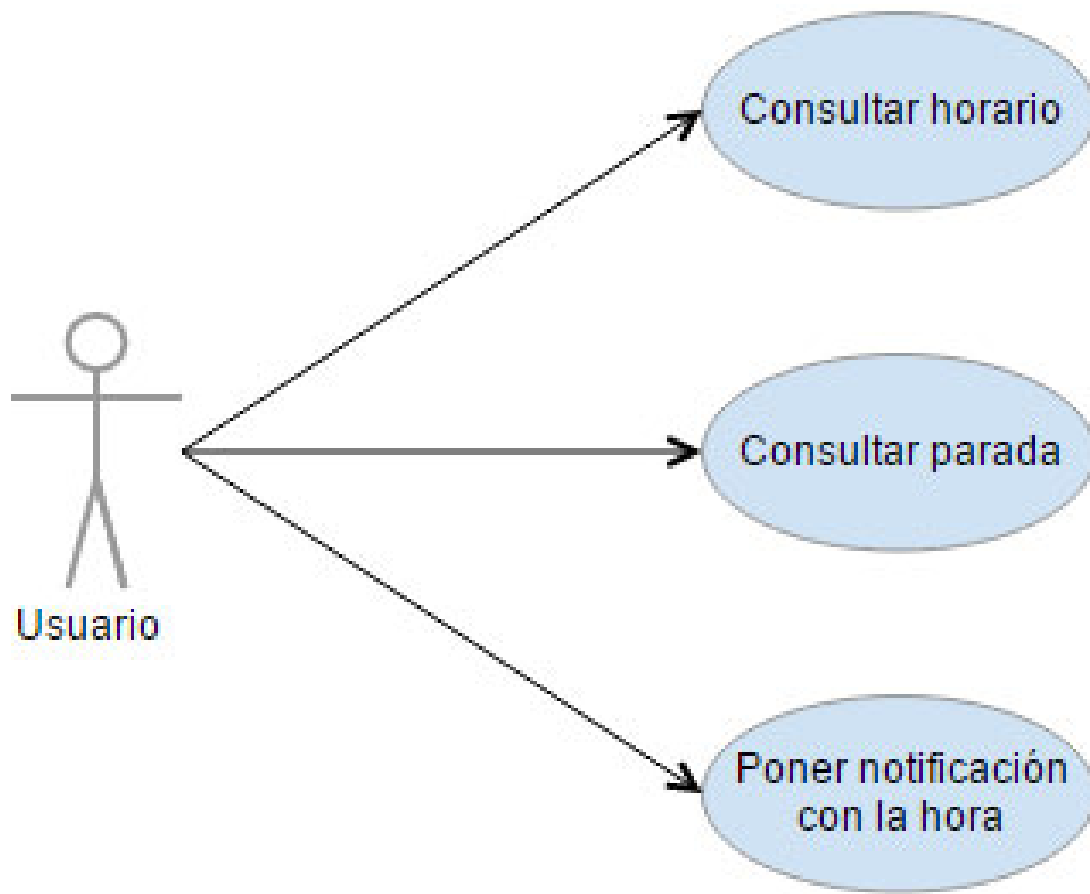
Además de los tableros, también se utilizan diagramas para conocer el estado del proyecto. En la figura anterior podemos ver las distintas fases (To Do en azul, In Progress en amarillo y Completed en verde).

Estos gráficos se utilizan para poder detectar fácilmente cuando se están produciendo cuellos de botella en el proyecto, por ejemplo en el lugar en el que señala la flecha roja, momento en el que los proyectos en progreso aumentan de cuatro a doce mientras que ninguno ha sido completado.

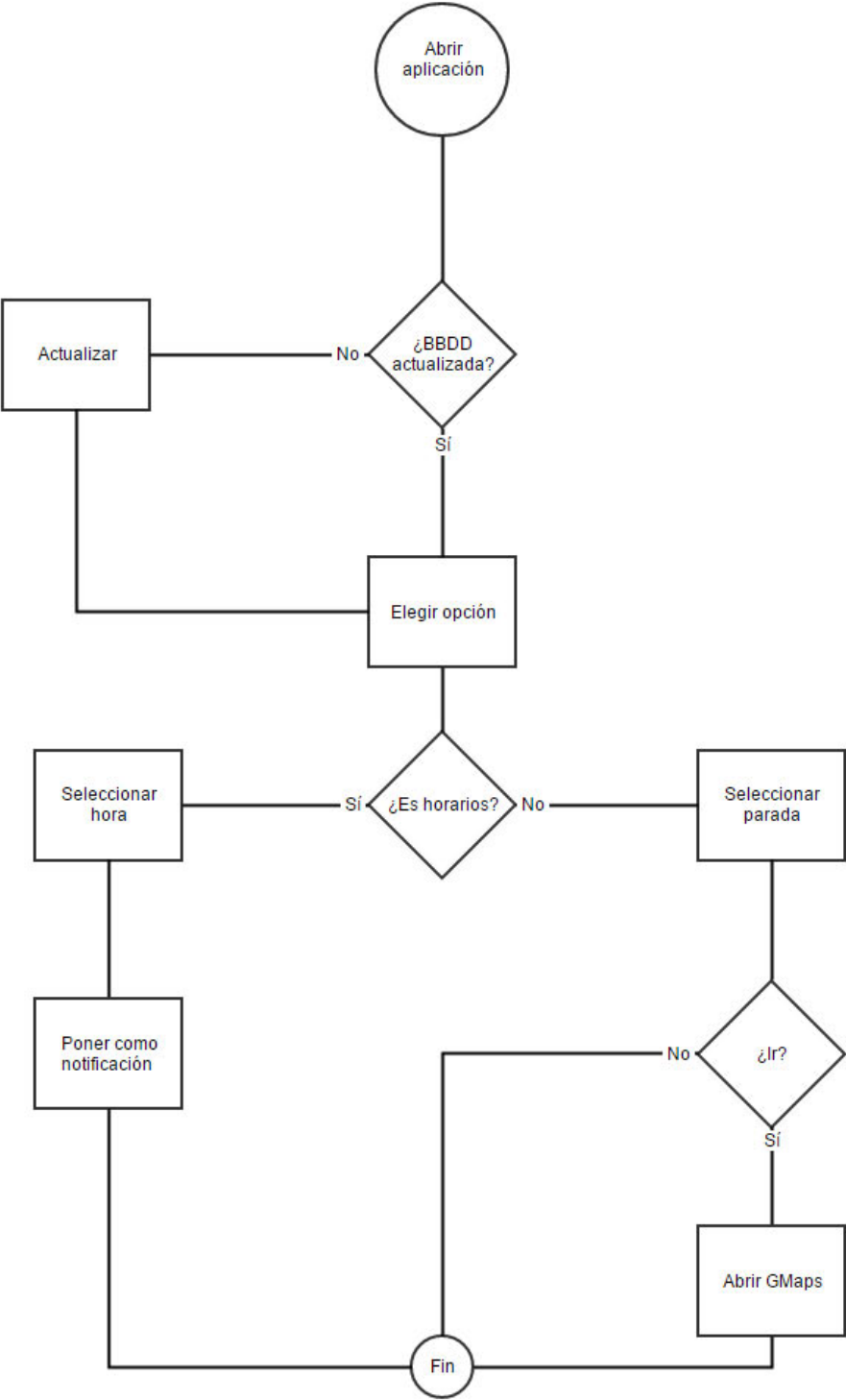
Una vez sabemos cuál es la metodología utilizada y conocemos sus principios, el primer paso es la especificación de los requisitos.

## Diagramas

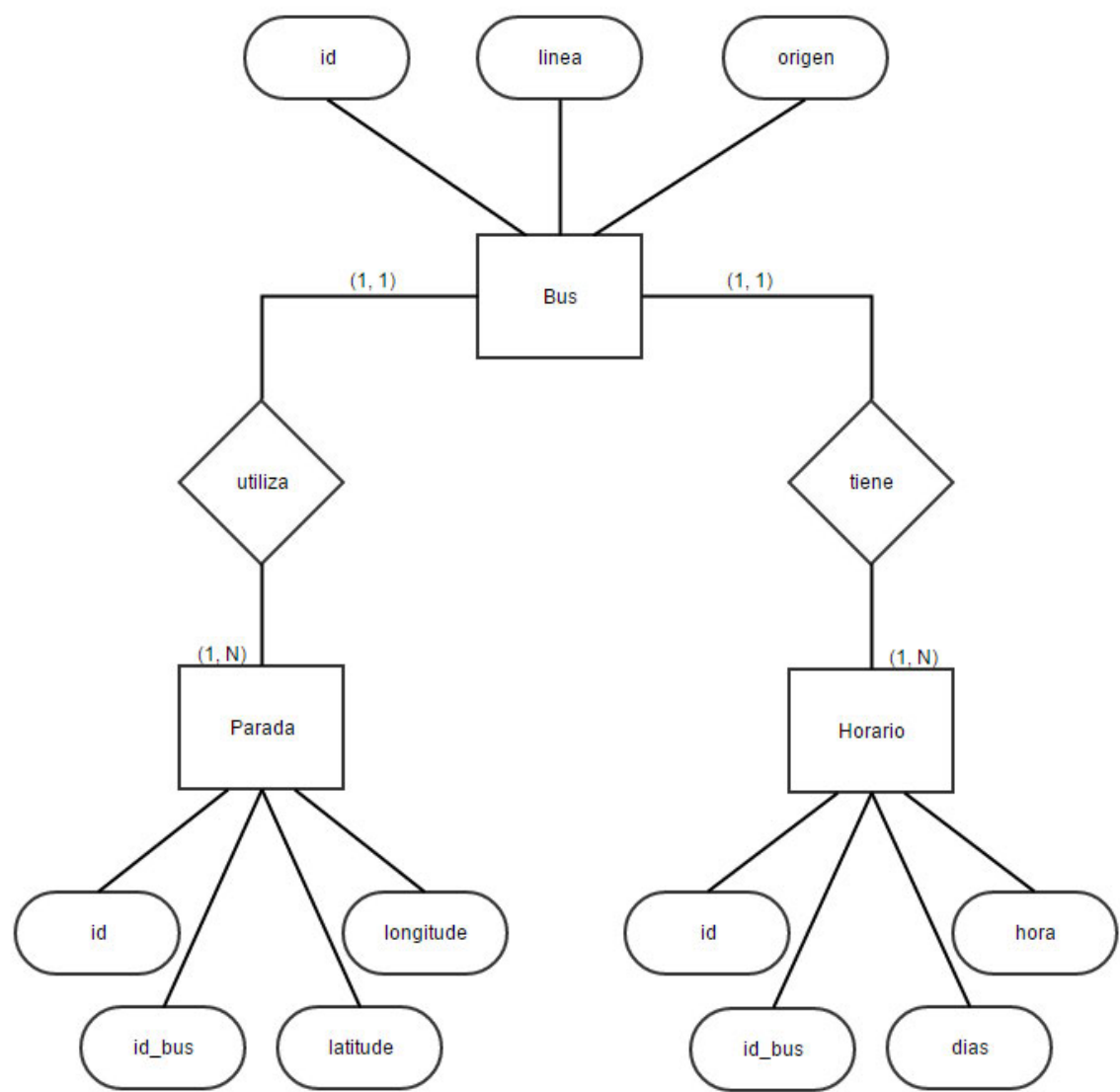
### Casos de uso



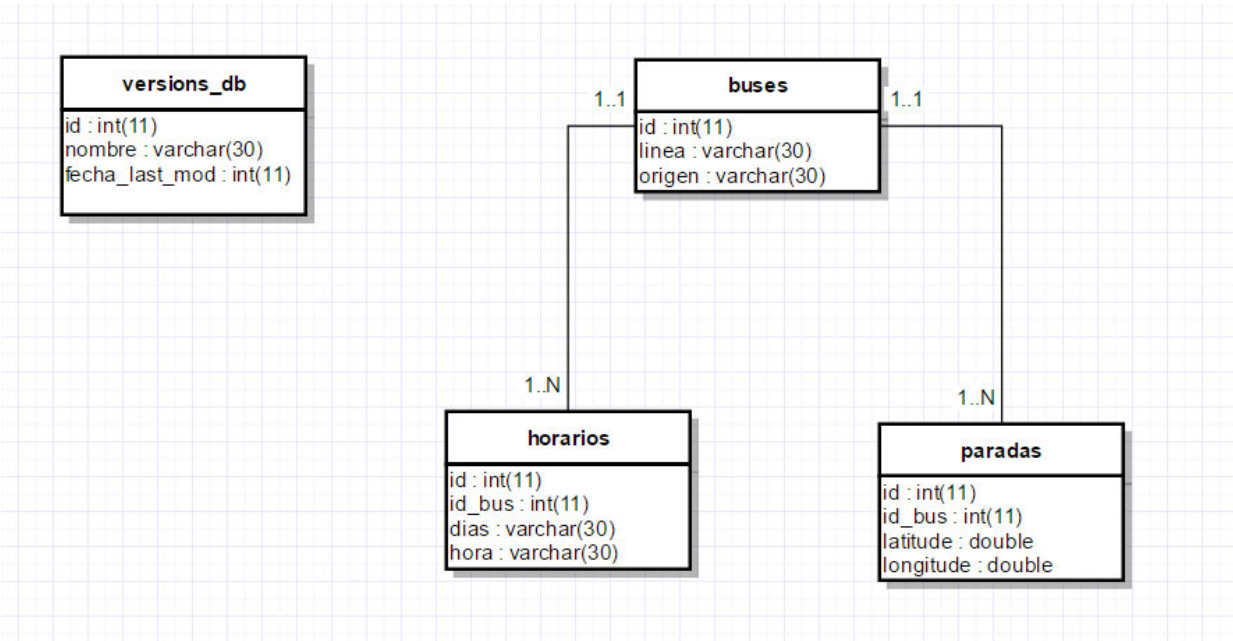
Flujo



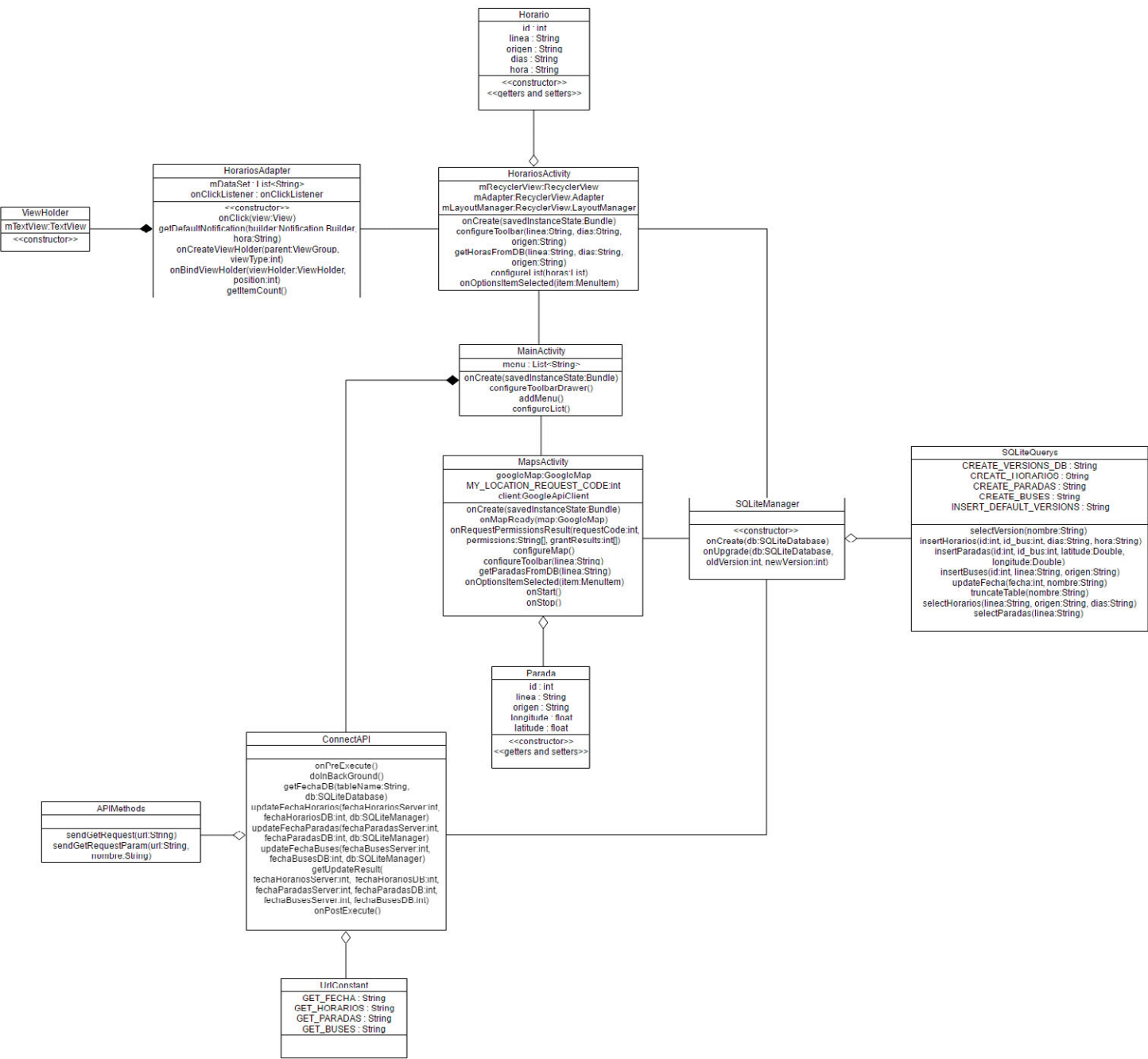
Modelo Entidad Relación



# Diseño base de datos

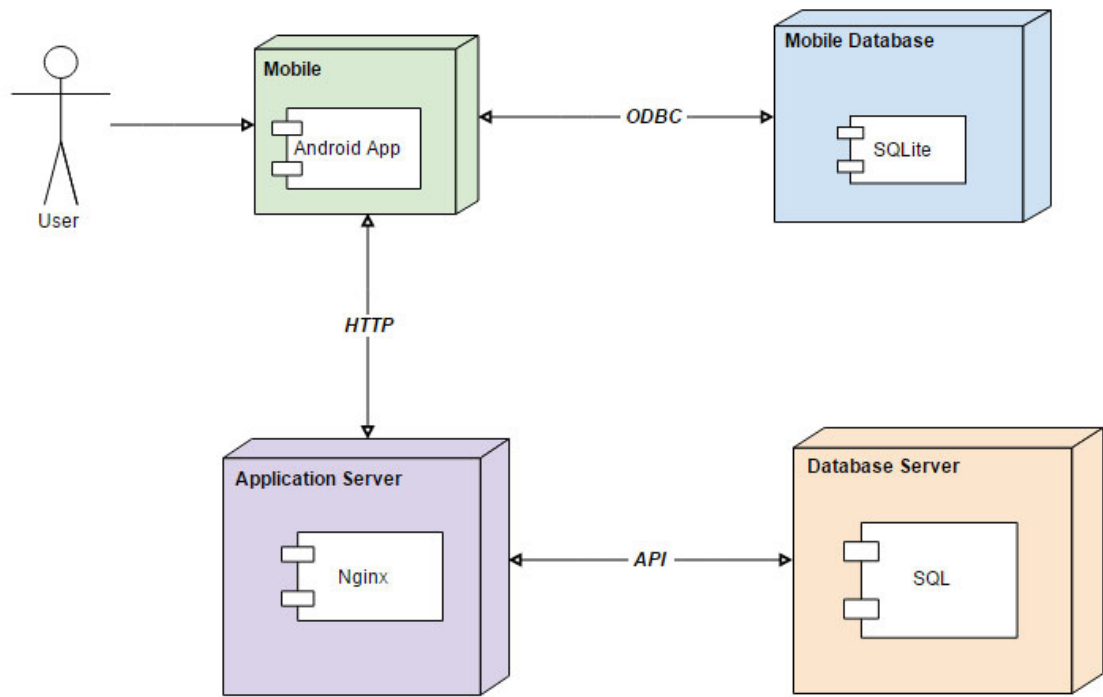


Clases





## Despliegue



## Especificación de requisitos

Los requisitos de un sistema son aquellos que describen los servicios que debe ofrecer y las restricciones a las que están asociados. El propósito de la especificación de requisitos es comportarse como un medio de comunicación entre los desarrolladores que trabajarán en el software y el cliente para el cual estamos trabajando.

Existen dos tipos de requisitos:

- Requisitos funcionales: son aquellos que describen como debe comportarse el sistema.
- Requisitos no funcionales: los cuales imponen restricciones sobre las posibles soluciones del sistema

Para el desarrollo de esta aplicación hemos extraído los siguientes requisitos:

Requisito	Funcional/No funcional	Descripción
RE1	No funcional	Debe ser intuitiva.
RE2	No funcional	Debe ser atractiva al uso.
RE3	No funcional	Debe ser escalable.
RE4	No funcional	Debe ser de fácil mantenimiento.
RE5	No funcional	Debe enviar y recibir datos de internet lo menos posible.
RE6	No funcional	Debe ser de fácil uso.
RE7	Funcional	Debe mostrar correctamente todos los horarios de las dos líneas de autobús.
RE8	Funcional	Debe mostrar correctamente en un mapa todas las paradas de ambas líneas.
RE9	Funcional	Debe actualizar los horarios y las paradas sin necesidad de descargar una actualización.
RE10	Funcional	Debe permitir poner una notificación con la hora de la salida del autobús seleccionado.

## Implementación de la metodología

Al haber desarrollado este proyecto una sola persona, necesitamos adaptar la metodología Kanban a un solo desarrollador con las ventajas e inconvenientes que esto supone.

Por un lado, al haber un solo desarrollador no existen problemas de sincronización en el proyecto y no se producirán retrasos entre las distintas fases al encargarse él mismo de todas, por lo que todo el tiempo empleado se invierte en la programación del proyecto.

Por otra parte, al encargarse una sola persona de todo el proyecto solo podrá existir una tarea en cada fase para que no haya problemas de sobrecarga. Esta tarea tendrá que ser desarrollada completamente antes de continuar con las demás.

El desarrollo de la aplicación se llevó a cabo implementando las mejoras necesarias para conseguir un producto funcional desde el primer momento, añadiendo en cada etapa una serie de funcionalidades definidas en la especificación de requisitos.

Tarea	Descripción	Requisitos
TA1	Recogida y tratamiento de información	RE7, RE8
TA2	Desarrollo de la vista donde se mostrarán los horarios	RE1, RE2, RE6, RE7, RE10
TA3	Desarrollo de la vista donde se mostrarán las paradas	RE1, RE2, RE6, RE8
TA4	Desarrollo de API	RE3, RE4, RE9
TA5	Diseño de base de datos	RE3, RE4
TA6	Conectar app a servidor	RE7, RE8, RE9
TA7	Alimentación de la base de datos	RE7, RE8
TA8	Desarrollo de la lógica de la aplicación	RE3, RE4, RE9

Una vez hemos definido nuestras tareas, el siguiente paso es definir las fases de nuestro proyecto. En este caso, tendremos las siguientes fases:

- Por hacer: las tareas de esta fase serán aquellas que están esperando para ser desarrolladas.
- En desarrollo: las tareas de esta fase serán aquellas que están siendo desarrolladas en ese momento.
- Realizadas: las tareas de esta fase serán aquellas que ya hayan sido desarrolladas.

Teniendo en cuenta las tareas definidas, las dividiremos en cuatro sprints de una semana de duración, haciendo una entrega al cliente del producto en la finalización de cada par.

Sprint	Tareas	Funcionalidad de la reléase
S1	TA1, TA2, TA8	Mostrará los horarios tomando los datos de un JSON
S2	TA3, TA8	Mostrará los horarios y las paradas tomando los datos de un JSON
S3	TA5, TA7	Mostrará los horarios y las paradas tomando los datos de la base de datos
S4	TA4, TA6, TA8	Mostrará los horarios y las paradas tomando los datos de la base de datos con actualizaciones automáticas

## Sprints

Debido a que la principal prioridad de esta aplicación es ser completamente funcional lo antes posible, comenzaremos haciendo una primera versión con todas las funcionalidades básicas, es decir, que nos muestre los horarios y las paradas de las líneas de autobús para, más adelante, realizar la aplicación del lado del servidor y diseñar e implementar la base de datos.

### Sprint 1

En este primer sprint realizaremos la recogida de información y el tratamiento de la misma para escribirla en formato JSON. Para ello hacemos una búsqueda en Google para localizar los horarios de estas líneas y calculamos las diferentes horas de salida de los autobuses. Esta búsqueda nos llevó a la obtención de las siguientes imágenes, con las que calcularemos todas las horas de los autobuses.

**411**

**Perales del Río - Madrid (Legazpi)**

45 min      20      Tiempo estimado de recorrido

**HORARIOS DE SALIDA DE PERALES DEL RÍO**  
(Caserío de Perales)

010316 (Vigente todo el año)

Lunes a viernes laborables						
De	5:30	a	20:50	cada	20	minutos
A	21:30	22:10	22:55	23:40	0:25	1:10
Sábados laborables, domingos y festivos						
De	6:00	a	0:45	cada	25	minutos
Noches de viernes y laborables vísperas de festivo						
A	1:55	2:30				
Noches de sábados y noches de domingos y festivos que sean vísperas de festivo						
A	2:30					

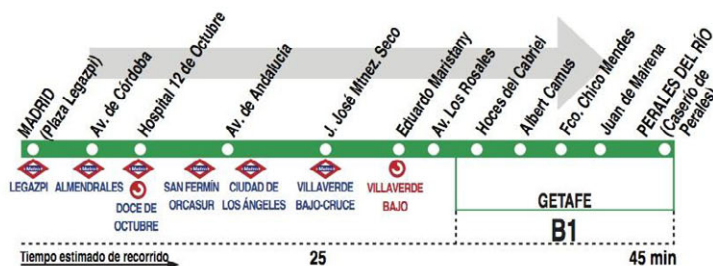
Notas: A efecto de horarios nocturnos se consideran los festivos nacionales y autonómicos.

**LV** LA VELOZ, S.A. Avenida del Mediterráneo, 49.  
28007 MADRID

**Tel: 91 409 76 02**

# 411

## Madrid (Legazpi) - Perales del Río



### HORARIOS DE SALIDA MADRID (Plaza Legazpi)

010316

(Vigente todo el año)

Lunes a viernes laborables						
	De	6:20	a	21:40	cada	20 minutos
A	22:15	22:55	23:40	0:25	1:10	1:50
Sábados laborables, domingos y festivos						
	De	6:50	a	1:35	cada	25 minutos
Noches de viernes y laborables vísperas de festivo						
	A	2:40	3:10			
Noches de sábados y noches de domingos y festivos que sean vísperas de festivo						
	A	3:10				

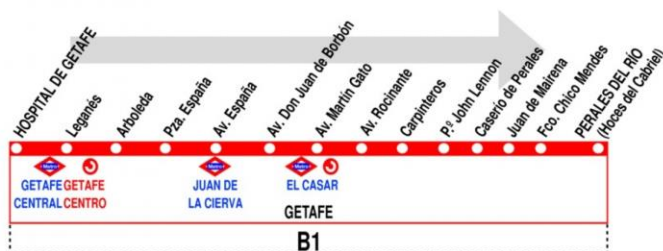
Notas: A efecto de horarios nocturnos se consideran los festivos nacionales y autonómicos.

**LV** LA VELOZ, S.A. Avenida del Mediterráneo, 49.  
28007 MADRID

Tel: 91 409 76 02

# 4

## Hospital - Perales del Río



### HORARIOS DE SALIDA DEL HOSPITAL DE GETAFE

010316

(Vigente todo el año)

Lunes a viernes laborables						
De	7:05	a	23:00	entre	30 - 35	minutos
Sábados laborables						
De	7:45	a	22:00	cada	45	minutos
		A	22:40			
Domingos y festivos						
De	9:15	a	22:45	cada	45	minutos
Noches de viernes, sábados y vísperas de festivo						
	A	23:25	0:45	2:05		

**AV** AVANZA INTERURBANOS, S.L.U.  
Ctra. de Toledo (A-42), p.k. 15,0. GETAFE 28905 MADRID

Tel: 91 695 24 70



# 4

## Perales del Río - Hospital

**HORARIOS DE SALIDA DE PERALES DEL RÍO**  
(Calle Hoces del Cabriel) (Vigente todo el año)

010316

Lunes a viernes laborables					
De	6:20	a	22:45	entre	30 - 35 minutos

Sábados laborables					
De	7:00	a	22:45	cada	45 minutos

Domingos y festivos					
De	8:30	a	22:45	cada	45 minutos

Noches de viernes, sábados y vísperas de festivo		
A	0:05	1:25

**AV** AVANZA INTERURBANOS, S.L.U.  
Ctra. de Toledo (A-42), p.k. 15,0. GETAFE 28905 MADRID

**Tel: 91 695 24 70**

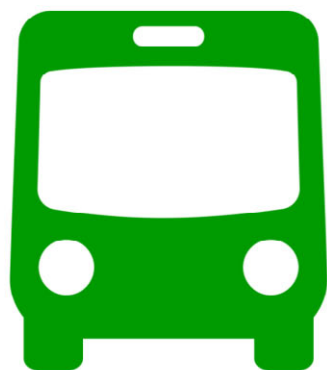
Además, utilizaremos nuestros conocimientos previamente adquiridos sobre las paradas que realizan estos autobuses en su recorrido junto con Google Maps para conseguir las coordenadas de cada una de ellas, quedándonos el siguiente JSON:

```
[
  {
    "linea": "L411",
    "origen": "Perales",
    "dias": "laborables",
    "hora": "05:30"
  },
  {
    "linea": "L411",
    "origen": "Perales",
    "dias": "laborables",
    "hora": "05:50"
  },
  {
    "linea": "L411",
    "origen": "Perales",
    "dias": "laborables",
    "hora": "06:10"
  }
],
```

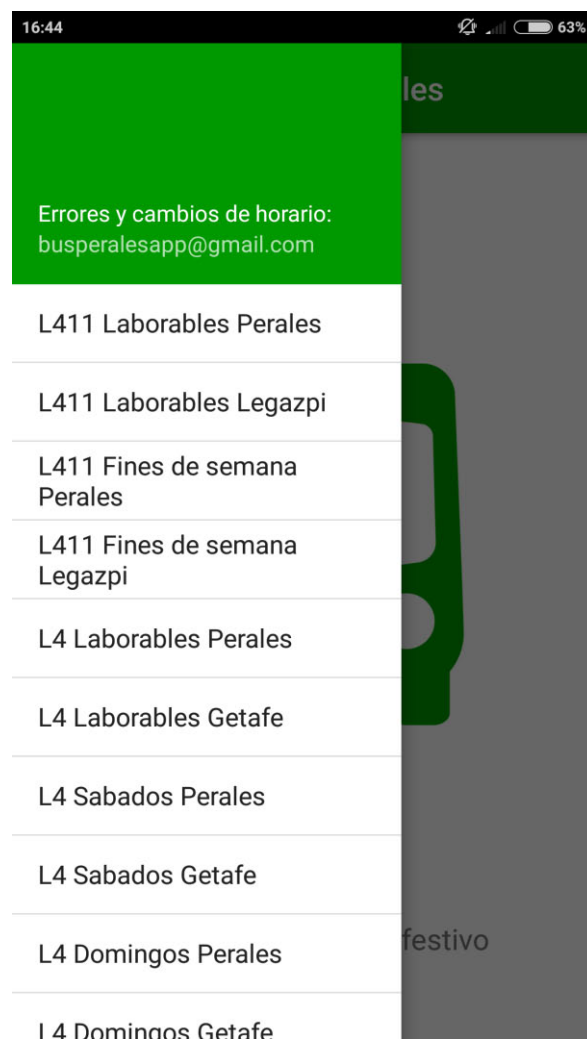
```
[
  {
    "linea": "L411",
    "origen": "Perales",
    "latitude": "40.312705321937955",
    "longitude": "-3.7391210365893857"
  },
  {
    "linea": "L411",
    "origen": "Perales",
    "latitude": "40.31086457084108",
    "longitude": "-3.736717777312042"
  },
  {
    "linea": "L411",
    "origen": "Perales",
    "latitude": "40.30889900233576",
    "longitude": "-3.7325898576381222"
  }
],
```

Para finalizar este sprint, realizaremos una primera versión de la aplicación en la que podremos consultar los horarios de los autobuses.

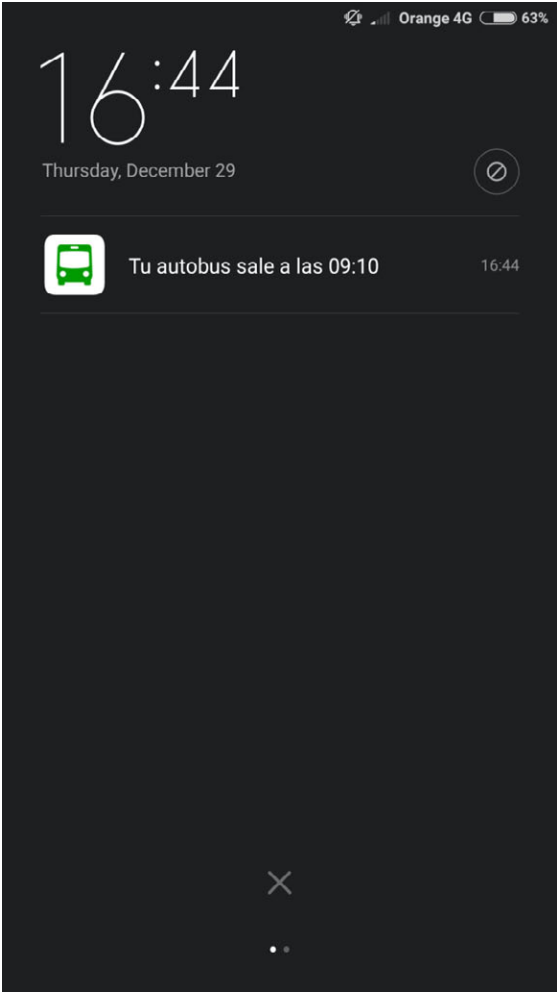
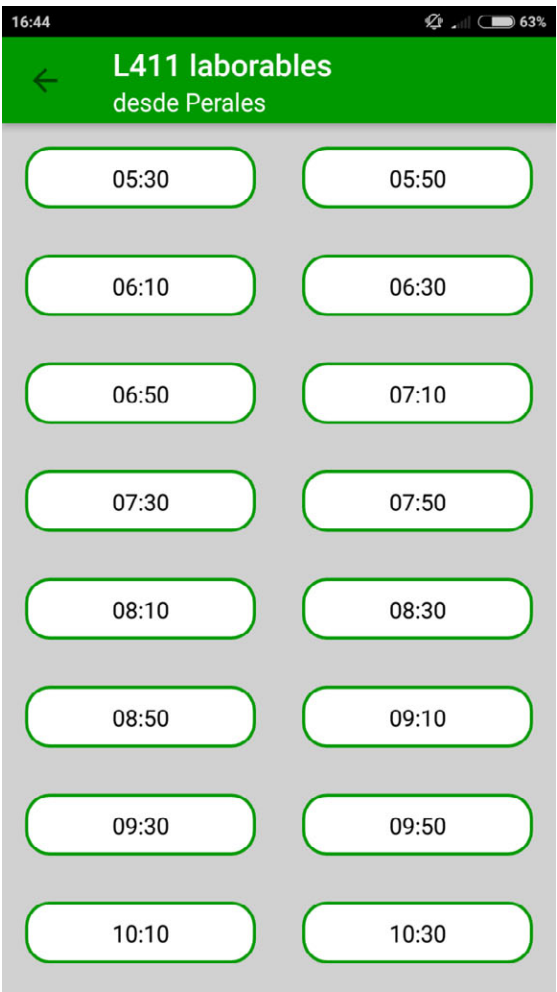
En las siguientes dos imágenes podemos ver, en la izquierda, la pantalla principal de la aplicación desde la que navegaremos a las demás pantallas y, a la derecha, el menú desplegable con las diferentes opciones entre las que, de momento, sólo aparecen los horarios.



\* Viernes y visperas de festivo  
\*\* Sabados



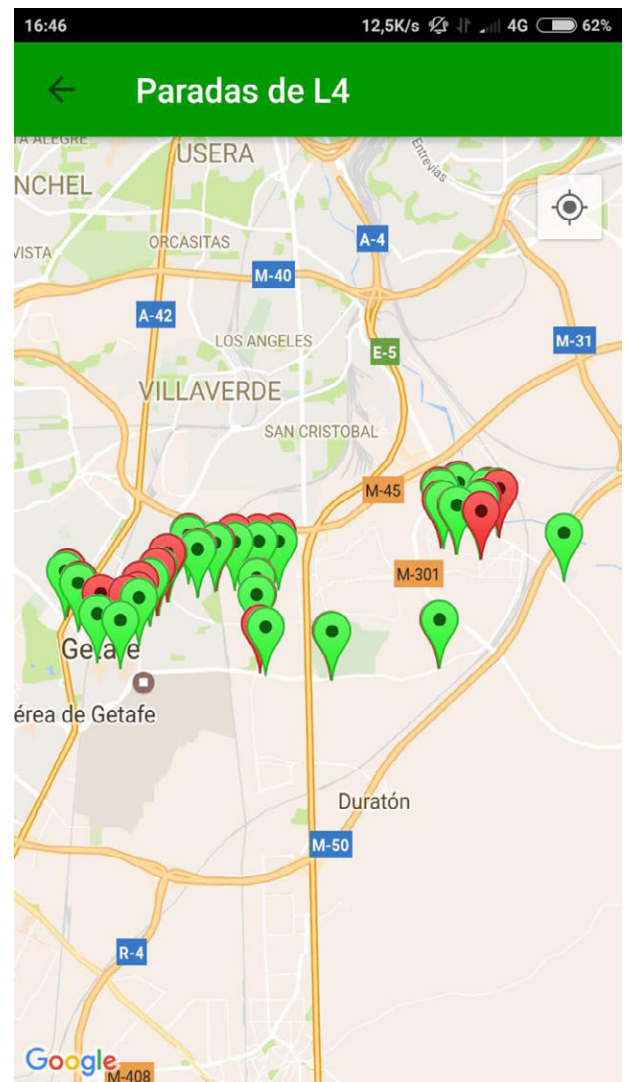
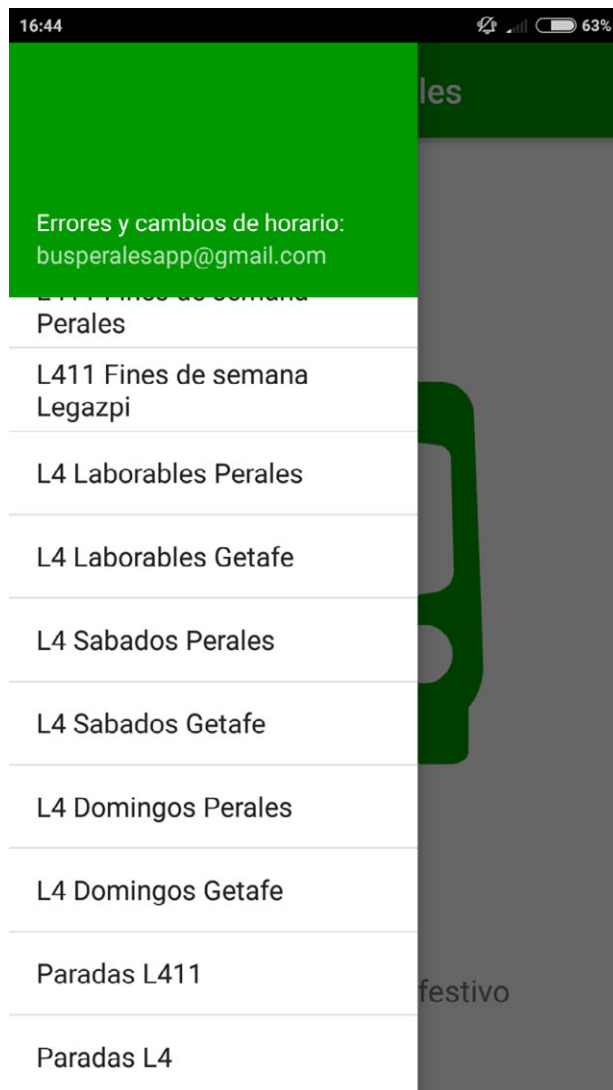
Estas otras dos imágenes muestran el resto del resultado del primer sprint; por un lado tenemos la vista con los horarios seleccionada desde el menú y por otro la notificación que se lanzará al clicar sobre cualquiera de estas horas,



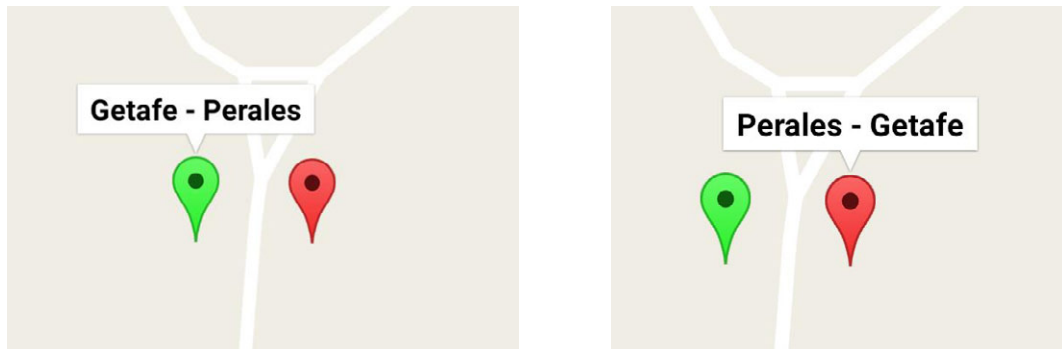
## Sprint 2

En este sprint realizaremos una segunda versión de la aplicación agregando las paradas a la misma. Éstas se visualizarán en un mapa de google que también cogerá nuestra ubicación y nos permitirá ver cómo llegamos hasta una parada en concreto utilizando la aplicación de Google.

En estas dos imágenes tenemos el menú desplegable con las nuevas opciones del mapa y la pantalla de Google Maps en la que hemos añadido markers para las diferentes paradas.



Además, para evitar confusiones con el color de los markers y el sentido del trayecto, hemos añadido un label que nos indica el origen y el destino del autobús cuando para por esa parada.



Al finalizar éste sprint tendremos una app con las funcionalidades básicas implementadas, lo cual nos permitirá realizar la primera entrega al cliente.

### Sprint 3

Durante este sprint, ya con la primera versión de la aplicación funcionando y entregada, nos dedicaremos a realizar el diseño, implementación y alimentación de la base de datos de la aplicación, tanto de la lado de la aplicación Android utilizando SQLite como del servidor, el cual tenemos alojado en Hostinger y nos permite utilizar phpMyAdmin y SQL para la gestión de la base de datos.

Nuestra base de datos se compone de cuatro tablas:

- Buses: esta tabla contendrá la información de los diferentes autobuses, diferenciándolos por línea y sentido. Es decir, por cada línea de autobús tendremos dos entradas en esta tabla: ida y vuelta.
- Horarios: esta tabla relacionará cada entrada de la tabla buses con sus horarios.
- Paradas: esta tabla relacionará cada entrada de la tabla buses con las coordenadas de sus paradas.
- Versions\_db: esta tabla contendrá una entrada por cada tabla que tenemos. Para cada una de ellas almacenaremos su nombre y un integer que muestra la versión de la última actualización.

En el próximo sprint desarrollaremos la lógica para que, cada vez que iniciemos la app, nuestra aplicación se conecte a la API y coja todas las entradas de esta tabla y de su homónima en la base de datos de la aplicación para compararlas y, de ser la del servidor mayor que la de la aplicación, actualizar en local la tabla que corresponda.

Con ésto lograremos que la aplicación actualice la base de datos cuando cambie en el servidor sin necesidad de descargar una actualización de la Play Store, mejorando la experiencia del usuario.

Además, cambiaremos la lógica de la aplicación para que tome los datos de la base de datos en lugar del JSON, pudiendo así eliminar éste de la app.

## Sprint 4

En este último sprint desarrollaremos una API Rest en PHP para poder acceder a la base de datos del lado del servidor y desarrollaremos los métodos necesarios en la app para lograr así que la base de datos de la aplicación se actualice automáticamente al modificar la del servidor.

En esta API únicamente implementaremos los métodos GET ya que serán los únicos que utilizaremos desde la aplicación y, a la hora de insertar y/o modificar datos utilizaremos phpMyAdmin, el cual tiene una interfaz que facilita su uso y los testaremos con Postman.



### View file dbConnect.php



```
1 <?php
2 define('HOST','mysql.hostinger.es');
3 define('USER','u456816571_admin');
4 define('PASS','');
5 define('DB','u456816571_db');
6 $con = mysqli_connect(HOST,USER,PASS,DB);
```



### View file getFechaLastMod.php



```
1 <?php
2 $nombre = $_GET['nombre'];
3
4 require_once('dbConnect.php');
5
6 $sql = "SELECT * FROM versions_db WHERE nombre=$nombre";
7
8 $r = mysqli_query($con,$sql);
9
10 $result = array();
11 $row = mysqli_fetch_array($r);
12 array_push($result,array(
13     "id"=>$row['id'],
14     "nombre"=>$row['nombre'],
15     "fecha_last_mod"=>$row['fecha_last_mod']
16 ));
17
18 echo json_encode($result);
19 mysqli_close($con);
```





## View file getAllBuses.php



```
1  <?php
2  require_once('dbConnect.php');
3
4  $sql = "SELECT * FROM buses";
5
6  $r = mysqli_query($con,$sql);
7
8  $result = array();
9
10 while($row = mysqli_fetch_array($r)){
11
12     array_push($result,array(
13         "id"=>$row['id'],
14         "linea"=>$row['linea'],
15         "origen"=>$row['origen']
16     ));
17 }
18
19 echo json_encode($result);
20
21 mysqli_close($con);
```

En la primera imagen podemos ver el archivo que nos permitirá conectarnos a la base de datos. El segundo nos devolverá la versión de la tabla con el nombre que le pasemos por parámetro al hacer la llamada y el tercero y último nos devuelve toda la tabla buses en formato JSON (además tenemos otros dos archivos que hacen lo mismo que éste último pero para las tablas horarios y paradas).

Al finalizar éste sprint tendremos la aplicación terminada y podremos hacer la segunda y última entrega al cliente.



## Metodología de pruebas

Para comprobar el correcto funcionamiento de nuestra aplicación hemos realizado una serie de pruebas de validación al terminar los sprints segundo y cuarto –con los que hacemos una entrega al cliente-. Para realizar estas pruebas hemos decidido realizar pruebas alpha y beta ya que así podremos comprobar si nuestro sistema se comporta de forma correcta en un uso normal del mismo.

Las pruebas alpha se llevan a cabo por el cliente con el desarrollador de observador registrando los errores y problemas de uso. Durante el desarrollo de estas pruebas realizaremos cambios en la base de datos del servidor para comprobar que la aplicación se actualizan debidamente

Las pruebas beta las realizan los usuarios finales sin la presencia del desarrollador de forma que se asemeje lo más posible al uso para el que está pensada la aplicación.

Para realizar este tipo de pruebas hemos seleccionado a un grupo de diez personas que forman parte del target de nuestra aplicación (personas que utilicen éstos autobuses de forma usual) con diferentes versiones de Android para así comprobar que el sistema se comporta como esperamos en todas las versiones de este sistema operativo. Estos usuarios utilizarán la aplicación durante un periodo de dos días y nos remitirán un informe con los errores que encuentren así como su experiencia con nuestra aplicación.

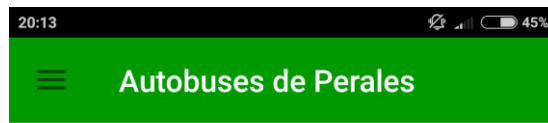
## Manual de usuario

Durante todo el desarrollo de la aplicación se ha buscado que el uso de la misma sea lo más simple posible de forma que pueda ser utilizada por los usuarios de Android menos habilidosos.

Actualmente, podemos darle dos usos principales, ver los horarios y poner una notificación o ver las paradas y cómo llegar a ellas.


### Ver horarios

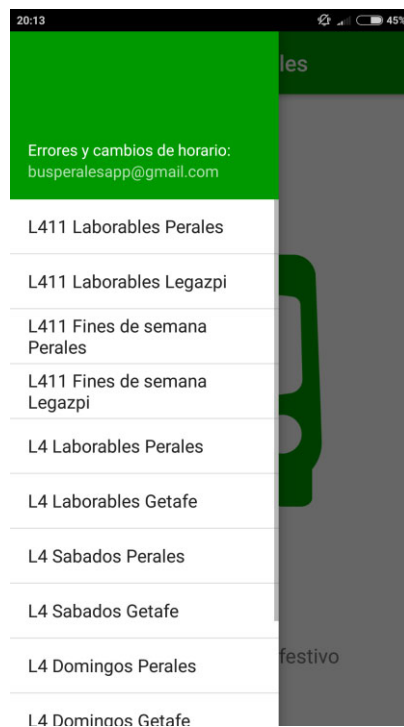
1. Abrir aplicación.



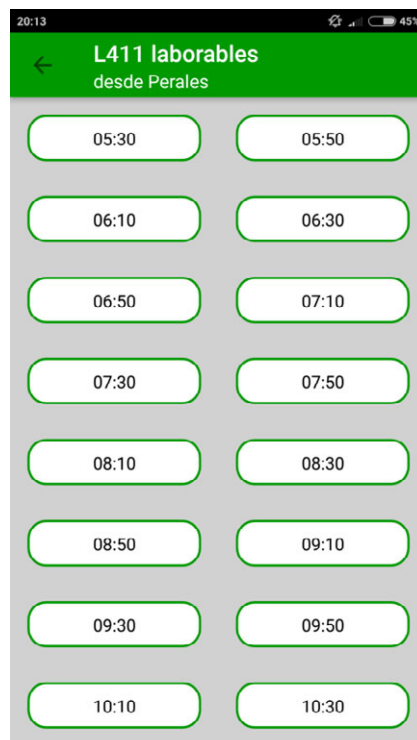
\* Viernes y visperas de festivo

\*\* Sabados

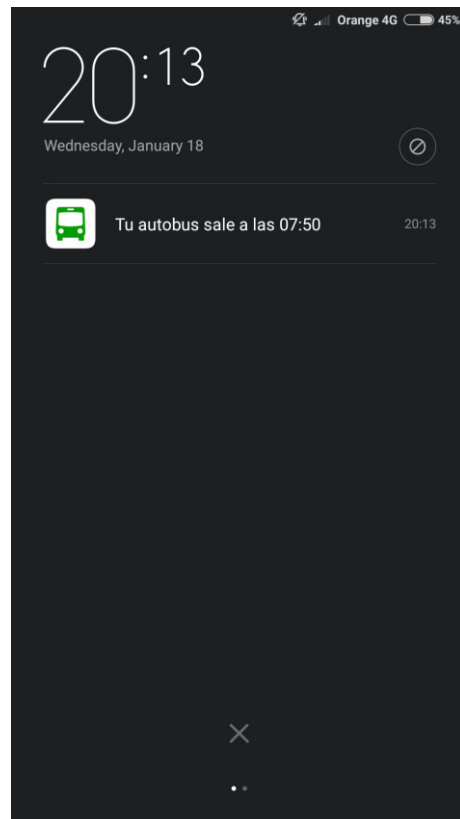
2. Abrir el menú haciendo tap sobre el icono  que se encuentra en la esquina superior izquierda o bien deslizando de izquierda a derecha desde el borde izquierdo.



3. Seleccionar el horario que deseamos ver.

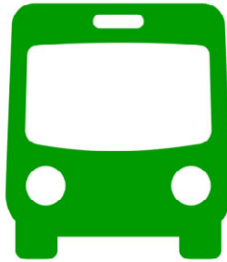
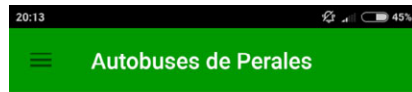


4. Si queremos poner una notificación, hacemos tap sobre la hora deseada.




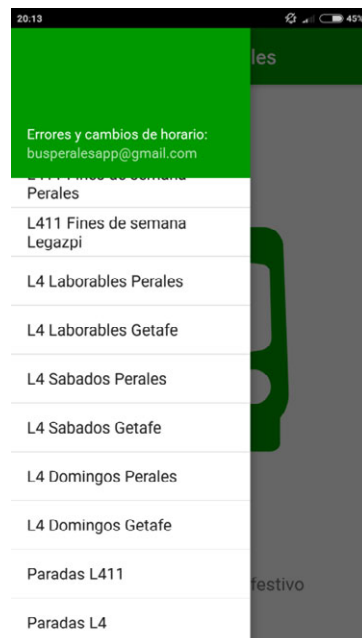
## Ver paradas:

1. Abrir aplicación.

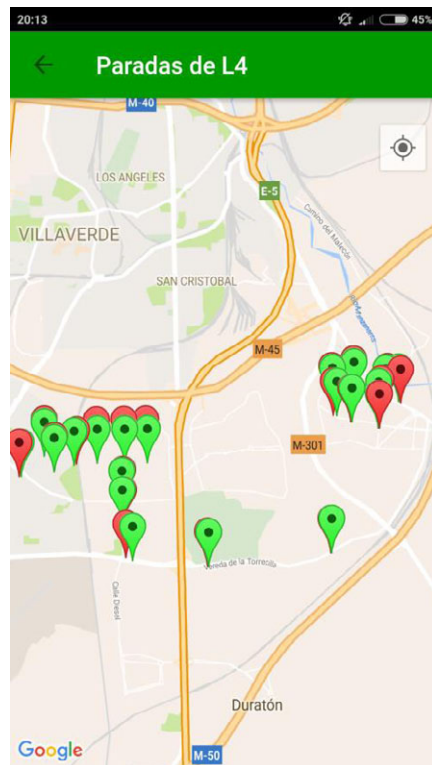



\* Viernes y visperas de festivo  
\*\* Sabados

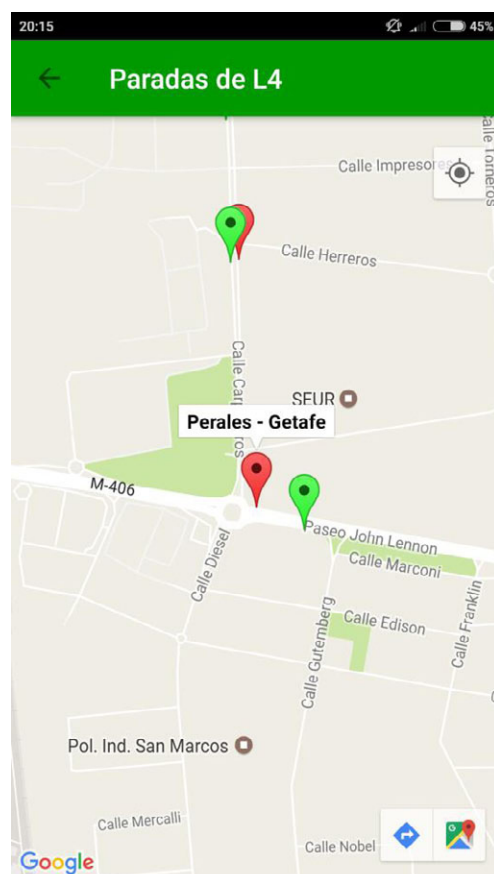
2. Abrir el menú haciendo tap sobre el icono  que se encuentra en la esquina superior izquierda.



3. Seleccionar las paradas que deseamos ver.



4. Si queremos saber qué dirección lleva el autobús en una parada, hacemos tap sobre ella. Si queremos saber cómo llegar a ella, pulsamos el icono  que aparece en la parte inferior derecha de la pantalla.



## Conclusiones y líneas de continuación

### Conclusiones

Durante el desarrollo de este proyecto se ha buscado desarrollar un sistema de información que permita a los usuarios consultar los horarios y las paradas de dos líneas de autobuses.

Para lograr este fin en el menor tiempo posible se dividió el trabajo en cuatro sprints, teniendo una release al finalizar cada par de forma que, después del segundo sprint la aplicación ya fuese funcional.

Además, se desarrolló una base de datos alojada en un servidor para poder actualizar la base de datos de la aplicación sin necesidad de descargar una actualización de la Google Play Store, de forma que los usuarios siempre tuviesen los horarios actualizados.

Durante el desarrollo de la aplicación se tuvieron diversos problemas de índole programática por la falta de conocimientos sobre Android, cuya resolución nos servirá para afrontar futuros proyectos a desarrollar para esta misma plataforma.

Al final el desarrollo del mismo teníamos como producto final un sistema fiable y robusto que aportaría a los usuarios de estos autobuses una información a la que a día de hoy no tenían acceso desde ninguna aplicación móvil o web.

Esta aplicación, pese a cumplir completamente los requisitos establecidos en este proyecto, aún puede ser mejorada –aunque en algunos casos hay que esperar a que terceras personas mejoren ciertos aspectos de estas líneas de autobús- que pasamos a describir a continuación.

## Líneas de continuación

Una de las principales mejoras que puede ser aplicada a nuestra aplicación es la forma de alimentar la base de datos del servidor ya que, de momento, por falta de medios hemos introducido los datos manualmente y las futuras actualizaciones las tendremos que hacer de la misma manera. Una vez haya un servicio que nos ofrezca la información que requerimos –esto es, horarios y paradas- de una forma fiable y actualizada podremos implementar un demonio en el servidor para que nuestra base de datos se actualice automáticamente al modificarse la de éste.

Además, una vez realicen el despliegue físico las empresas que gestionan estas líneas de autobuses y creen un servicio donde se pueda consultar el tiempo de espera en cada parada podremos implementarlo en nuestra aplicación.

Por último, se podría implementar un sistema de gestión de usuarios para permitirles guardar sus paradas y horas de autobús más frecuentadas, logrando así mejorar la experiencia del usuario.



## **Bibliografía**

<https://developer.android.com/training/index.html?hl=es>

<http://www.sgoliver.net/blog/curso-de-programacion-android/indice-de-contenidos/>

<https://gradle.org/>

<https://play.google.com/store>

<https://www.adslzone.net/2016/01/27/windows-phone-en-minimos-ios-sobrevive-y-android-crece-en-espana/>

<http://www.redtransporte.com/madrid/autobuses-interurbanos/411-madrid-getafe.html>

<http://www.redtransporte.com/madrid/autobuses-urbanos/l-4-getafe.html>

<http://www.restapitutorial.com/>

<http://stackoverflow.com/>

